# Error assessment and code speed-up for SOLPS-ITER

Developed within the framework of WP-CD 2016

Manual version 2.0

M. Baelmans[1], P. Börner[2], M. Blommaert[1], K. Ghoos[1], and W. Dekeyser[1]

[1]KU Leuven, Department of Mechanical Engineering, 3001 Leuven, Belgium
[2]Institute of Energy and Climate Research (IEK-4), FZ Jülich GmbH, D-52425 Jülich, Germany

May 4th, 2017

**Executive Summary**

In 2016 the work on SOLPS code speed-up in terms of accuracy is finalized by its implementation in the SOLPS-ITER version. The procedure essentially consists of three subsequent steps. Every step provides an additional level of information on simulation results, its related errors and optimal use of numerical parameters. The first step is the reference user run. In this step, the averaging procedure is used to achieve simulation results with reduced statistical noise. Therefore, it is recommended to be used in all cases. In addition, variances are monitored to estimate the statistical error. It is shown in [1] that this averaging procedure introduces a small bias. In this first step, however, the value of this bias is not yet known. Nevertheless, it becomes already apparent from residual evaluation that increased accuracy is achieved.

In order to achieve optimal numerical parameters given a required accuracy of simulation results for a specific case and its related grid size, the bias needs to be assessed. To this end, runs with a different number of Monte Carlo particles are needed. Based on theoretical error reduction rates [1, 2] for statistical error and bias, the optimal numerical parameters can be set to minimize CPU for a desired accuracy. The averaging procedure enables us to drastically reduce the statistical error in the solution within a reasonable computational time. As a result of this, we now can go even one step further and assess the discretization error, i.e. the error related to the finite grid size.

The procedure as described in this manual is applied to a slab test case first and subsequently tested for a Deuterium only ITER F12 case, a C-MOD Deuterium only case and a C-MOD case with Deuterium and Boron.

This report provides a practical user manual for the averaging procedure to be used to determine numerical parameters for code speed-up. Information on underlying theoretical and numerical studies can be found in [1–3].

## 1 Introduction

This report focuses on providing the necessary documentation to serve its users. In the following section the basic run with averaging procedure is documented. It is implemented with the intention for general use, meaning for use by all SOLPS-ITER users.
The expert level to assess the errors and to speed-up the code can be performed now following the methodology described in [3]. By assessing the different contributions an optimal set of numerical

parameters (# Monte Carlo particles and # of iterations) can be obtained. It is recommended to do this for each new case or whenever strong changes in plasma parameters occur. The procedure allows to obtain accurate results in a much faster way. This opens perspective to evaluate discretization errors.
This first version of the manual focuses on general user information to enable averaging. This manual will be further expanded in the future based on the user's experience and needs.

# 2 Running SOLPS-ITER with averaging procedure

## 2.1 Introduction

The averaging procedure is the essential first step to improve the code results and to assess the statistical error. Essentially nothing changes with respect to the iteration process, the run, itself. The tool provides an improved interpretation of the simulation results via averaging over iterations only. This iteration averaged value is not fed back to the iteration procedure. From a theoretical point of view, it is shown [1] that an averaging procedure over successive iteration steps of a coupled Finite Volume (FV) / Monte Carlo (MC) results in a solutions with smaller statistical error than the one associated with the instantaneous plasma solution. By averaging a (very small) finite particle bias is introduced. In the following we will distinguish between an instantaneous plasma solution and an iteration averaged solution.

- The **instantaneous plasma solution** is the solution at some point in the iteration process; it is this value that thus far always was considered as the solution of the set of governing equations.
- The **iteration averaged solution** is computed by averaging instantaneous plasma solutions achieved over subsequent iterations.

Former studies revealed that the statistical error of iteration averaged solutions decreases inverse proportional to the amount of iterations $I$ over which is averaged ($\propto 1/\sqrt{I}$). Similarly, the error also decreases with increasing number of MC particles $P$ ($\propto 1/\sqrt{P}$). On the other hand the bias introduced through averaging is observed to be smaller than the statistical error of the instantaneous field for all applications studied thus far and decreases inverse proportional with $P$.
It is clear that the former statements for statistical errors hold in case the plasma fields used for averaging are in statistically steady state. Therefore we distinguish between an initial phase and the statistically steady state phase.

- **Statistically steady state** means that the iteration averaged value is no longer changing over iterations.
- In contrast, during the **initial phase**, iteration averaged solutions clearly show a transient behaviour

In order to limit disk space needed for averaging, we distinguish between data needed for monitoring and data needed for establishing accurate results and related estimation of the statistical error. For monitoring purposes there is a need now for determining the end of the initial phase. This is provided by batch averages of a limit set of properties which are written to the .nc file. More specifically we distinguish batch averages from running averages.

- **Batch averaging** means that an average is taken over a given number of iterations. When the number of iterations is reached a next batch average is calculated over the subsequent number of iterations.
- In contrast, we will refer to **running averages** when averaging, once started, is continued over all subsequent iterations, even in case a run is restarted.

Running averages are used once the initial phase is over and statistically steady state is reached. At this instance the procedure starts with providing running averages of all plasma properties and source terms. To enable continuation of the averaging procedure even after restarting the run, the plasma file with running average values and running average sources, as well as their variances, will be stored. Thus, it is guaranteed that the averaging procedure can be continued. Again along the run, it is interesting to observe how well the averaged values are obeying the governing equations. This is assessed by monitoring the residuals of the averaged plasma solutions. To this end running averages of plasma profiles and sources are saved at intermediate iteration numbers to keep track of the convergence. The different aspects in the averaging procedure are illustrated in figure 1.

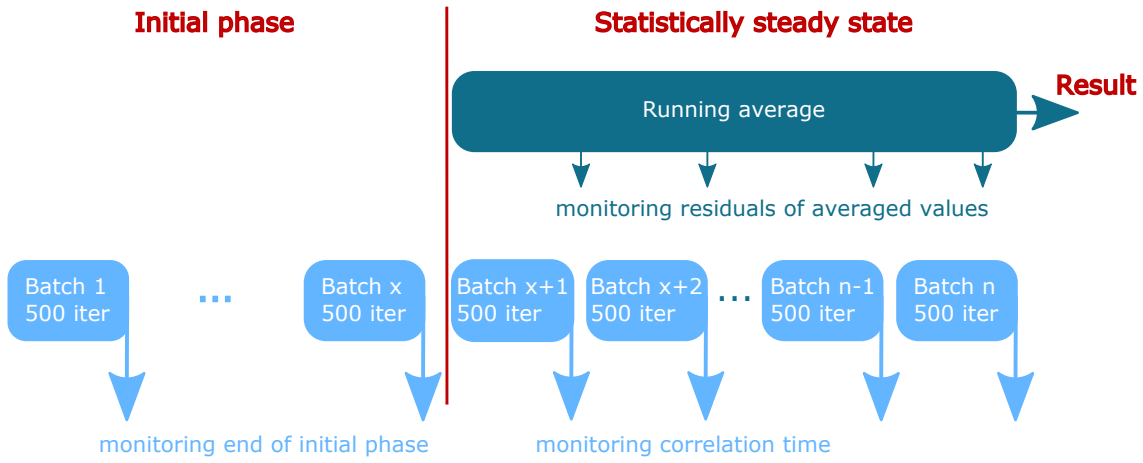


**Figure 1:** Schematic of the averaging procedure

In the following the procedure to practical use iteration averaging is described. An overview of the new variables introduced for the averaging procedure can be found in section 2.5.

## 2.2 Step 1a: Batch averaging in the initial phase

When starting a new test case, envisaging the averaging procedure, we first need to check when the initial phase is over. To this end we monitor batch averages of specific quantities. Only after this initial phase the solution reaches its statistically steady state and the averaging procedure to reduce the statistical noise can start. The quantities to be monitored in the initial phase are the same type of quantities as used for monitoring the instantaneous values. A subset of these quantities, now with its batch averaged values, is made available in the **b2time.nc** file by default (i.e. when b2mndt_av_batch_all $>$ 0, see also section 2.3). These batch averages are less noisy than the instantaneous ones and as such are better suited for visualising transient behaviour.
In addition to the regular input of your case add the input parameters as listed in table 1.

Note 1: The computation of batch averages of default quantities is not limited to the initial phase, (assuming b2mndt_av_ntim_batch $>$ 0). These quantities are also used for monitoring and eventually statistical error assessment in the statistically steady state.
Note 2: It is possible to extend the predefined variables. This requires further implementation work (cfr. set of predefined variables for the regular b2time.nc file). A full list of monitored instantaneous quantities can be found in the SOLPS-ITER manual, Appendix E.
Note 3: For expert use, it is possible to set b2mndt_av_batch_all = 1. In this case, batch averages of 2D fields of the plasma states and their variances are provided in ./batch_av/batch_av.XXXX every

| Variable | Value | Description |
|---|---|---|
| **in b2mn.dat** | | |
| **b2mndt_av** | 0 (Def.) | No computation of the running average |
| **b2mndt_av_ntim_batch** | e.g. 500 (Def.) | When larger than 0, number of iterations used to compute batch averages |
| **b2mndt_av_batch_all** | 0 (Def.) | Standard output for batch averages |
| **in input.dat** | | |
| **ninitl** | $< 0$ | parameter to ensure full randomness of the seed to be done in block 7 for each stratum in $1^{st}$ line of integers $2^{nd}$ position |

**Table 1:** Input parameter values for initial phase

b2mndt_av_ntim_batch iterations. Be aware that this will consume a lot of disk space!

### 2.3 Step 1b: Determine transient phase

To determine the transient phase, batch averages of (selected) quantities are stored in b2time.nc during the run (see step 1a). Determine with visual inspection whether the file is in its statistically steady state. When you reach this phase, you start the running average as described in step 1c. The following list of quantities are available for batch-monitoring:

| Variable | Description |
|---|---|
| **nesepm_av, nesepi_av, nesepa_av** | separatrix electron density at outer midplane, inner and outer target |
| **nemxip_av, nemxap_av** | max electron density at inner and outer target |
| **tesepm_av, tesepi_av, tesepa_av** | separatrix electron temperature at outer midplane, inner and outer target |
| **temxip_av, temxap_av** | max electron temperature at inner and outer target |
| **tisepm_av, tisepi_av, tisepa_av** | separatrix ion temperature at outer midplane, inner and outer target |
| **timxip_av, timxap_av** | max ion temperature at inner and outer target |
| **posepm_av, posepi_av, posepa_av** | separatrix electric potential at outer midplane, inner and outer target |
| **pomxip_av, pomxap_av** | max electric potential at inner and outer target |

**Table 2:** Default monitored plasma parameters

These quantities should be plotted against the batch time, which is stored in the array **batchsa** in the b2time.nc file. **batchsa** is the total amount of simulated time (in seconds) at which the batch average is computed (the last iteration of the batch). Thus, the batch averages can easily be plotted on a same axis as the other time dependent quantities (nesepm, nesepi, etc.). The latter are plotted against the **timesa** array (also in seconds). The difference between **batchsa** and **timesa** is that **timesa** is defined at time steps where the standard time dependent quantities are written, while **batchsa** is defined at time steps where batch-averages are written.
To visualize the batch averaged quantities use *2dt_av nesepm_av nesepi_av* etc. The script is available in $SOLPSTOP/scripts . For the slab case the separatrix density values at the outer midplane, inner and outer target are shown in figure 2 (left), whereas the evolution of the maximum density at inner

and outer target is shown in figure 2 (right). The instantaneous residuals obtained with the *res_all* are shown for illustration in figure 3.

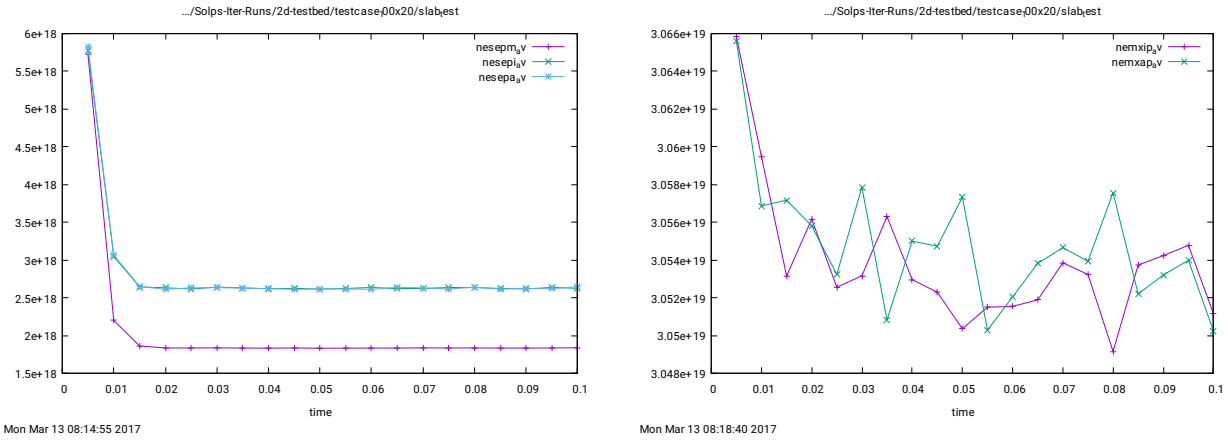


**Figure 2:** Sample of batch averaged quantities for a slab test case to monitor the end of the initial phase (20 batches of 500 it.): separatrix density values at the outer midplane, inner and outer target(left), maximum density at inner and outer target (right)

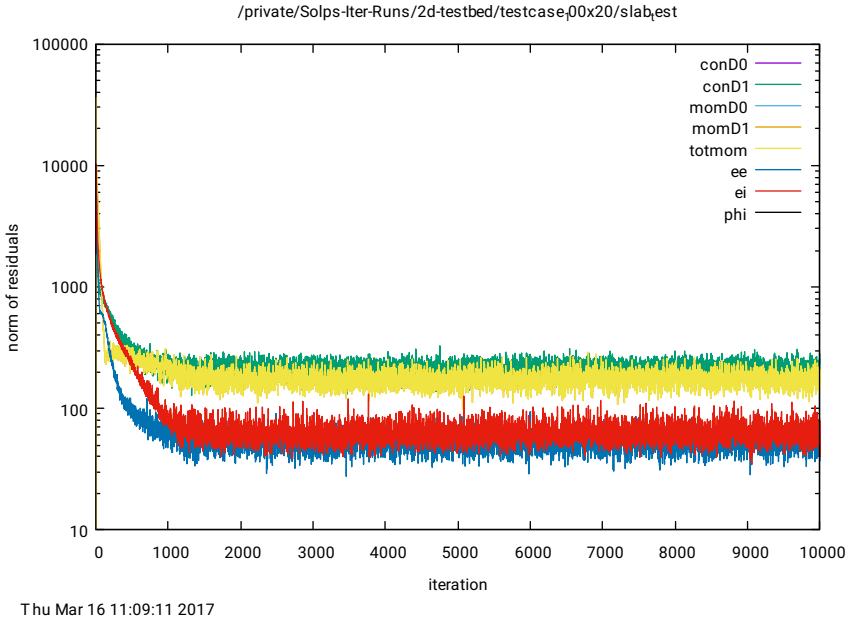


**Figure 3:** Instantaneous residuals in intial phase

## 2.4 Step 1c: Compute averaged solution and its residuals

Once the initial phase is over, the effective running averaging procedure is started. During the averaging procedure in steady state, the averaged plasma state values, the averaged sources and their respective variances are saved into a file called **b2favere** at the end of the run. The averaging procedure starts from 0 in case no initial averaging file **b2faveri** is available. For monitoring the residuals of the running averaged profiles the simulation results are saved at regular times (controlled by

**b2mndt_av_ntim_run**). This data is stored in **./run_av/run_av.XXXX** files.
The running averaging procedure is activated by adding the following input to your case:

| Variable | Value | Description |
|---|---|---|
| **b2mndt_av** | 1 | Computation of the running average |
| **b2mndt_av_continue** | 1 (Def.) | Continuously perform running averages |
| **For monitoring purposes** | | |
| **b2mndt_av_ntim_run** | e.g. 1000 (Def.) | Number of it. for writing running averages |
| **b2mndt_av_ntim_batch** | e.g. 500 (Def.) | Number of it. for computing batch averages for monitoring purposes only |
| **b2mndt_av_batch_all** | 0 or 1 | Standard or expert output for batch averages |

**Table 3:** Input parameter values in statistically steady phase

Once the run_av files are produced the residuals can be monitored by calling the script **res_av**, which will produce the resall_D plot. Furthermore, it will create a b2ftrace_av file which can be used for additional plots by renaming it to b2ftrace.
The final plasma simulation results can now be found in b2fstate_averaged. The EIRENE simulation results need to be obtained from this plasma as an initial state and using a high number of particles to reduce statistical noise. To this end the parameters listed in table 4 need to be set in the b2mn.dat file. A residual plot for the slab test case is shown in figure 4. The number of iterations here indicate the number of batches (in this case to be multiplied with b2mndt_av_ntim_batch = 500). It should be noted that the residuals of the instantaneous plasma values (see figure 3) are very noisy and are several orders of magnitude larger than the ones obtained through averaging (figure 4). The averaging procedure already results in low residuals after the first batch and further decrease for some equations as the statistical error is further reduced. Continuation of the run lead to a further reduction of the electron energy residuals, meaning that the statistical error is further reduced by continuing the averaging procedure.
Further, when the stagnation level for the residuals is reached, it is now dominated by the bias and gives a first estimate of the bias error introduced by the finite amount of particles used in the EIRENE run. This stagnation level can be lowered by using more EIRENE particles. In this particular example it is clear that the residual for the energy equation is still sensitive to statistical noise rather than by the finite sampling bias, whereas continuity, momentum and ion energy reach their saturation by the bias at a much earlier stage.

| Variable | Description |
|---|---|
| **in b2mn.dat** | |
| **b2mndr_ntim = 2** | Only two iteration steps are required to give the averaged profile to EIRENE |
| **b2mndt_rxf = 0.0** | No changes to the averaged plasma profiles are made |
| **in input.dat** | |
| **block 7, $1^{st}$ row of real numbers, $2^{nd}$ position** | Multiplicative factor to enhance simultaneously all the numbers for particles released at the different strata |

**Table 4:** Input parameters to obtain the final plasma and neutral particle solution

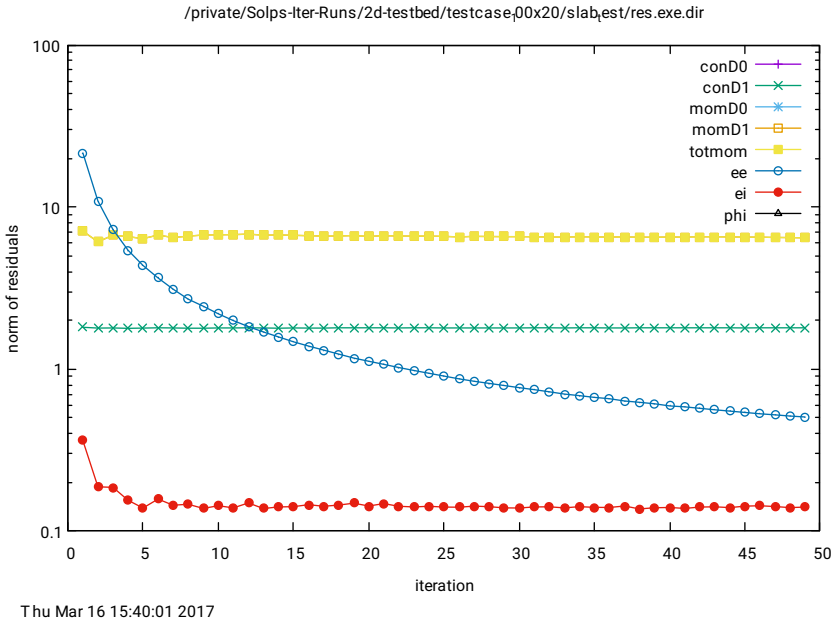


**Figure 4:** Residuals of the running averaged solution for a slab test case

Note 1: The importance of starting the averaging procedure only after the initial phase is illustrated in figure 5. Here residuals of the running averages where averaging was started right from the beginning, i.e. from the initial plasma solution obtained with a neutral fluid model. The residuals will finally also drop to a small value, the computational effort is however significantly increased!

Note 2: In order to guarantee also a backup file for the averaging procedure, running averages and variances are stored at regular times. This procedure is similar to the one foreseen for the instantaneous plasma parameters. The plasma data is stored in plasmastate_av.XXXX .

Note 3: For the slab test case, the statistiscal error can easily be made small enough to enable symmetry checks for this symmetrical test case. Thus, this test case brings additional verification for new implementations in the coupling between B2.5 and EIRENE. As an example comparison of density and ion temperature profiles obtained with B2.5-EIRENE (SOLPS-ITER) are shown in figures 6 and 7 respectively, for the symmetric test slab case. Relative differences between left and right target are in the order of $10^{-5}$ to $10^{-4}$.

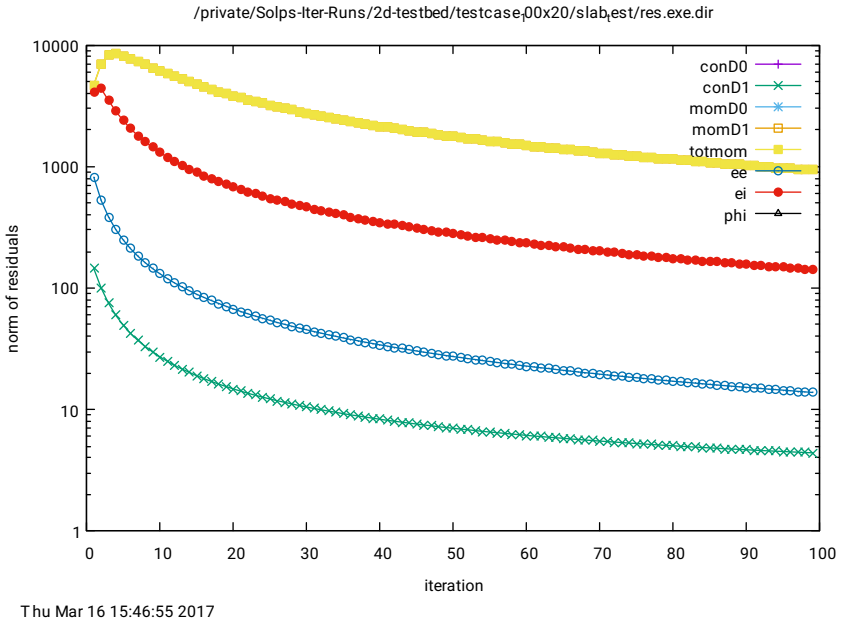


**Figure 5:** Residuals of the running averaged solution for a case where the averaging was started too early

### 2.5 Description of the additional input variables introduced

The new variables introduced for the averaging procedure are tabulated in table 5.

## 3 Optimal parameters for code speed-up - expert level

In order to achieve optimal numerical parameters to minimize the sum of statistical error and finite particle bias, the user needs to derive the governing constants for the error reduction rates for a specific quantity of interest, e.g. the overall heat flux to the outer target. To this end the user needs to run the SOLPS-ITER case with three different values for the number of particles launched in EIRENE. These values are preferentially taken at least a factor 10 apart, but can be performed with a reduced amount of particles rather than with an increased number. Thus the bias is more easily provoked. Once the numerical constants (cfr. the ones listed in ( [3], table 3) are extracted from the nc file, the optimal value can be computed with the Matlab-script *optimalP.m*. For further information the reader is referred to [3].

## Acknowledgement

This work has been carried out within the framework of the EUROfusion Consortium and has received funding from the Euratom research and training programme 2014−2018 under grant agreement No 633053. The views and opinions expressed herein do not necessarily reflect those of the European Commission. The work of K. Ghoos is sponsored by the Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT- Vlaanderen).

## References

[1] K. Ghoos, W. Dekeyser, G. Samaey, P. Börner, D. Reiter, and M. Baelmans, “Accuracy and convergence of coupled finite-volume / Monte-Carlo codes for plasma edge simulations of nuclear

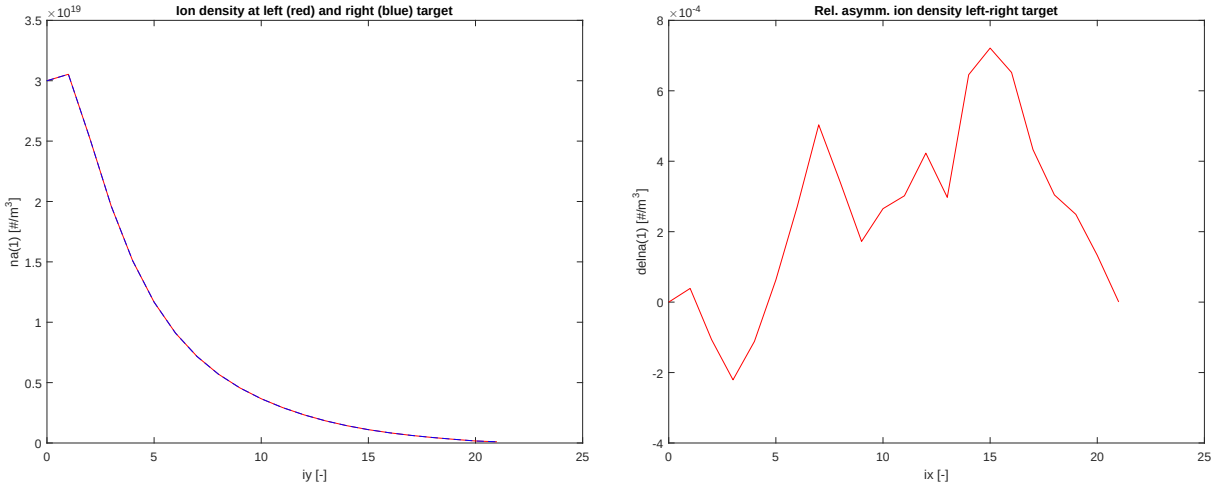


**Figure 6:** Ion density profile (left) and its relative asymmetry (right) for the slab test case

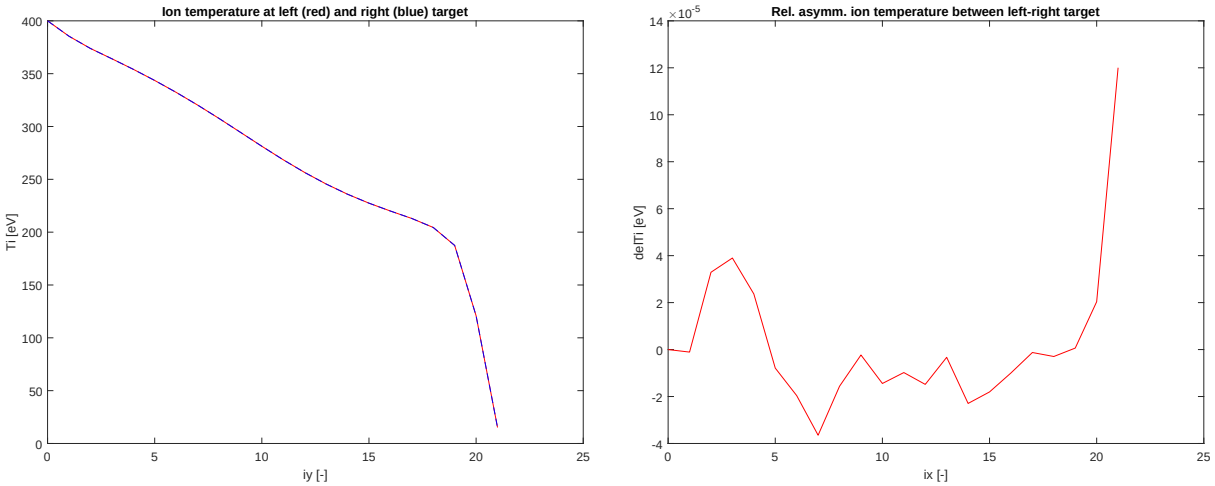


**Figure 7:** Ion temperature profile (left) and its relative asymmetry (right) for the slab test case

fusion reactors",” *Contributions to Plasma Physics*, vol. 56, no. 6-8, pp. 616–621, 2016. Plasma Edge Theory Conference, Nara, Japan.

[2] K. Ghoos, W. Dekeyser, G. Samaey, and M. Baelmans, “Accuracy and convergence of coupled finite-volume / Monte-Carlo codes for plasma edge simulations of nuclear fusion reactors,” *Journal of Computational Physics*, vol. 322, pp. 162–182, 2016. 2nd Frontiers in Computational Physics: Energy conference, Zurich, Switzerland.

[3] M. Baelmans, P. Börner, K. Ghoos, and G. Samaey, “Efficient code simulation strategies for b2-eirene,” *Nuclear Materials and Energy*, vol. 1-6, 2016.

| Variable | Value | Description |
|---|---|---|
| **b2mndt_av** | **(Def. 0)** | **Controls computation of running average** |
| | 0 | No computation of running average |
| | 1 | Computation of running average |
| **b2mndt_av_continue** | **(Def. 1)** | **Controls continuation of running average** |
| | 0 | Restart of running averages is explicitly set |
| | 1 | Running averages start from data previously stored in files **b2fstate_run_av** and **b2fstate_run_var** by copying the data to the respective **b2fstati** files if data is not available averaging is restarted |
| **b2mndt_av_ntim_run** | **(Def. 1000)** | **Number of it. for writing running average** |
| | 0 | No writing of running average |
| | $> 0$ | Number of iterations for writing running average |
| **b2mndt_av_ntim_batch** | **(Def. 500)** | **Number of iterations used for batch averages** |
| | 0 | No batch averaging |
| | $> 0$ | Number of iterations used for batch average |
| **b2mndt_av_batch_all** | **(Def. 0)** | **Controls output quantities for batch averages** |
| | 0 | Default set output with limited monitoring points |
| | 1 | Entire 2D fields of batch averages and variances for $n_e$, $T_e$, ... are written to directory ./av_batch/av_batch.XXXX. Expert use only! – requires lot of storage! |
| **b2mndr_av_read** | **(Def. 0)** | **Controls initial plasma (instantaneous or averaged)** |
| | 0 | Initial plasma is instantaneous plasma |
| | 1 | Initial plasma is based on averaged state |

**Table 5:** New variables introduced for averaging and error assessment